Experiments with Computer Vision Methods for Fall Detection

Zhong Zhang, Eric Becker, Roman Arora, and Vassilis Athitsos
Computer Science and Engineering Department
University of Texas at Arlington
Arlington, Texas, USA

ABSTRACT

The goal of a fall detection system is to automatically detect cases where a human falls and may have been injured. A natural application of such a system is in home monitoring of patients and elderly persons, so as to automatically alert relatives and/or authorities in case of an injury caused by a fall. This paper describes experiments with three computer vision methods for fall detection in a simulated home environment. The first method makes a decision based on a single frame, simply based on the vertical position of the image centroid of the person. The second method makes a threshold-based decision based on the last few frames, by considering the number of frames during which the person has been falling, the magnitude (in pixels) of the fall, and the maximum velocity of the fall. The third method is a statistical method that makes a decision based on the same features as the previous two methods, but using probabilistic models as opposed to thresholds for making the decision. Preliminary experimental results are promising, with the statistical method attaining relatively high accuracy in detecting falls while at the same time producing a relatively small number of false positives.

Categories and Subject Descriptors

H.3.1 [**Content Analysis and Indexing**]: Indexing methods; H.2.8 [**Database Applications**]: Data Mining; H.2.4 [**Systems**]: Multimedia Databases

General Terms

Experimentation

Keywords

Fall detection, Non-parametric model, Gaussian model

1. INTRODUCTION

Over the past few years, several researchers have focused on designing robust algorithms to understand and recognize human actions in videos [7, 6]. This is a challenging research area, and progress in that area can be directly applicable in real-world applications. As an example, smart monitoring systems that recognize human actions and events can be helpful for improving the safety of elderly people and patients living unaccompanied at home.

A useful functionality of such a home monitoring system is fall detection, i.e., the ability to automatically detect cases where a human falls and may have been injured. In this paper, we describe three different computer vision methods to recognize fall actions in videos. The first method makes a decision based on a single frame, simply based on the vertical position of the image centroid of the person. The second method makes a threshold-based decision based on the last few frames, by considering the number of frames during which the person has been falling, the magnitude (in pixels) of the fall, and the maximum velocity of the fall. The third method is a statistical method that makes a decision based on the same features as the previous two methods, but using probabilistic models as opposed to thresholds for making the decision.

The results that we obtain illustrate the promise of applying computer vision methods to home assistance and protection, as we obtain relatively good results with observations from a single camera and relatively few training data. At the same time, our goal is to significantly improve the accuracy and robustness of the proposed method, by collecting significantly larger amounts of training data, and by studying the incorporation of sensors and additional cameras into the environment.

2. RELATED WORK

Several approaches have been proposed for fall recognition (see [12] for a recent review). The majority of existing approaches are not pure vision-based, but instead use input from a variety of sensors to make a decision[2, 3, 9, 11, 15]. The most commonly used sensor is the accelerometer. Chen et al. [4] use a low-cost and low-power MEMS accelerometers to detect the fall, and they have compared the observed acceleration of falling backwards, sitting and walking. Zhang et al. [15] embed a tri-axial accelerometer in a cellphone, and use KFD(Kernel Fisher Discriminant) and the k-nearest neighbor rule for classification. Lindemann et al. [9] set accelerometers behind the person's ear, and propose 3 thresholds to trigger a fall detection: acceleration in the xy-plane higher than 2g; velocity of all spatial components right before the impact higher then 0.7 m/s; and the sum of acceleration of all spatial components higher than 6g. In [3, 2], the authors propose fall algorithms based on thresholds on both signals from a tri-axial accelerometer and a biaxial gyroscope. In [1], three different sensors: a 3D Time-Of-Flight range camera, a wearable MEMS accelerometer and a microphone, have been used to detect the height of the person, an acceleration peak during the fall event, and acoustic data respectively. Thresholds for the centroid height and acceleration peak are

PETRA '10, June 23-25, 2010, Samos, Greece.

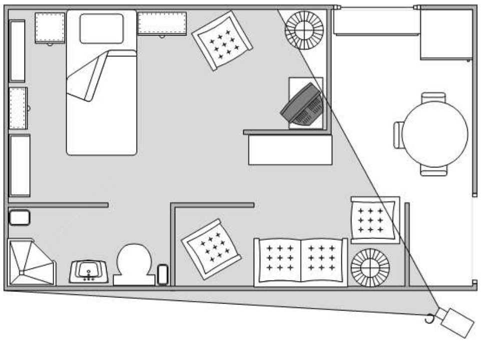

Figure 1: Simulated home environment

used to determine whether the current action is a fall or not, and Hidden Markov Models (HMMs) are used to analyze the acoustic scene. Jansen et al. [5] obtain a variety of features using 3D camera technology, and pass these features to a simple classifier to decide whether there is a fall or not.

Vision-based methods have also been utilized for fall detection. Wu [14] found that the both the vertical and the horizontal speed of falls are 2-3 times higher than other normal actions, and the increase in vertical and horizontal magnitude usually occurrs simultaneously in falls. Rougier et al. [13] track the the head and uses the 3D head trajectory to detect falls. Nait-Sharif et al. [10] combine a learned models of spatial context and a tracker to detect falls in videos.

3. SCENE SETUP

The experiment data for this paper come from experiments run in the Heracleia Human Centered Computiing Laboratory at the University of Texas at Arlington. In this lab, a simulated apartment has been set up with a kitchen, den, bedroom, and bathroom. A video camera was set up at one corner of the apartment, and was set to monitor the bedroom and television area, and the bathroom of the apartment, as can be seen in Figure 1.

4. THRESHOLD-BASED METHODS FOR FALL DETECTION

In this section we describe two threshold-based approaches for fall detection. All methods use only information obtained from a single camera, as described in Section 3.

4.1 A Method Based on Vertical Position

Since our method is geared towards fall detections in a home enviroment, we make the assumption that the main source of motion in the scene is the person that is being monitored. To detect motion, we have used a simple method based on frame differencing. More sophisticated background subtraction methods, using, e.g., Mixtures of Gaussians(MoGs) [8], can be used instead, but the simple frame differencing method has worked sufficiently well for our experiments so far.

Frame differencing works as follows: let $I(x, y, i)$ denote the intensity value at pixel (x, y), at the i-th frame. By comparing $I(x, y, i)$ with $I(x, y, i-z)$ and $I(x, y, i+z)$, we compute a binary motion indicator value $M(x, y, i)$ that is set to 1 iff motion occured at pixel (x, y) at the i-th frame. Motion indicator value $M(x, y, i)$ is defined using the following equations:

$$I_1(x, y, i) = \begin{cases} 1 & if \quad |I(x, y, i) - I(x, y, i - z)| > T \\ 0 & otherwise \end{cases} \qquad (1)$$

$$I_2(x, y, i) = \begin{cases} 1 & if \quad |I(x, y, i) - I(x, y, i + z)| > T \\ 0 & otherwise \end{cases} \qquad (2)$$

$$M(x, y, i) = I_1(x, y, i) \wedge I_2(x, y, i) \qquad (3)$$

In the above equations, T is a system-specific threshold. Once $M(x, y, i)$ has been computed for all pixels of frame i, connect component analysis is used to identify the largest connected component of pixels (x, y) with values $M(x, y, i) = 1$. We call that component the *person* region. If t and b are the highest and lowest row coordinates of the person region, the row centroid $H(i)$ of the person at frame i is defined as:

$$H(i) = t/2 \qquad (4)$$

The first method we have tried for fall detection simply applies a threshold h on $H(i)$. When the row centroid H is less than h, we consider that a fall event has happened. Obviously, the choice of threshold has a big impact on how well this method works. In the experiments section we provide experiment results with different thresholds.

4.2 Using Duration, Change in Position, and Maximum Speed

The second method is based on the observation that, when there is a fall, the row centroid of the person decrease rapidly over a few frames. We can use the duration of the fall in frames, the total change in the row centroid during the fall, and the maximum speed (maximum rate of change of the row centroid) during the fall as features for determining whether a fall has occurred.

As in Equation 4, we denote by $H(i)$ the human row centroid. We denote by k the length of the fall (i.e., the number of frames in which the row centroid of the person keeps moving lower in the image), by m the maximum change in the row centroid from one frame to the next during the fall, and by d the total change in the row centroid. The following simple pseudocode determines k and m.

Once k, m, and d have been computed for the current frame i, determining whether a fall has occrred not is done by applying thresholds k_1, k_2, d_1, m_1, and t_1, as follows:

$$\text{fall} = \begin{cases} 1 & \text{if}(k_1 \leq k \leq k_2) \wedge (d \geq d_1) \wedge m(\geq m_1) \wedge (H(i) \leq t_1) \\ 0 & \text{otherwise} \end{cases} \qquad (5)$$

Again, the performance of this method depends heavily on the choice of thresholds. In the experiments we provide results with different settings for the thresholds.

5. USING A STATISTICAL MODEL

The previous two methods depend heavily on manually chosen thresholds. Choosing the thresholds in an optimal manner can be challenging. Furthermore, an infinitesimal change in the fall process can make a feature value be over or under a certain threshold,

```
Input: current frame index $i$.
Output: the duration of the fall $k$, the max row centroid
change $m$ and the total change in the row centroid $d$;

$max\_fall\_length = 20$;
$k = 1$;
$m = 0$;
while $k < max\_fall\_length$
  if $H(i-k) < H(i-k+1)$
    break;
  end
  if $H(i-k) - H(i-k+1) > m$
    $m = H(i-k) - H(i-k+1)$;
  end
  $k = k + 1$;
end
$d = H(i-k-1) - H(i)$;
```

Algorithm 1. Computing the duration of the fall, the max frame-by-frame row centroid change, and the total change in the row centroid.

thus making a big difference in the output of the system. In this section we describe a more principled method, that uses a statistical model automatically constructed from training data.

We perform training in a user-independent manner using cross-validation. We split our recorded videos into four sets, such that no human subject appears in more than one of those sets. Then, each set is used, in turn, as a test set, on which we apply the statistical model constructed using the remaining three sets as training data. Fall events (start and end frames) are manually annotated in all videos, and this information is used during training and also to measure the accuracy on the test data.

At each frame i, four features are extracted: as in Section 4.2, we use fall duration, denoted as k, the maximum frame-by-frame change of the row centroid, denoted as m, and the total change of the person's row centroid in the fall process, denoted as d. We also use the lowest row centroid in the fall process, denoted as c.

5.1 Training Positive and Negative Models

We assume that the four features are conditionally independent, given the class of the event, where the class can be "positive" (fall) or "negative" (non-fall). We use non-parametric models to simulate the probability distributions features for the negative samples. As the set of training positive samples (i.e., fall samples) only contains six samples at each round of cross-validation, we model the feature distributions for the positive samples as gaussian distributions. By assuming that features are conditionally independent, we only need to compute a mean and standard deviation for each feature separately, and six samples are sufficient for that purpose.

In order to describe how to build the nonparametric distributions for the negative samples, we use feature m (maximum frame-by-frame change of row centroid) as an example. Every negative sample has a corresponding value of m. Let $m1$ and $m2$ be the smallest and largest value of m among the negative samples. We discretize the interval $m1$ and $m2$ into N_m bins, and compute the frequency of values in each bin among the negative samples. By normalizing the frequencies to sum up to 1, we have computed a discretized probability function $P(m|\text{non-fall})$. Similarly, we compute such discretized histogram-based probabilities for each of the four features.

5.2 How to Make a Decision

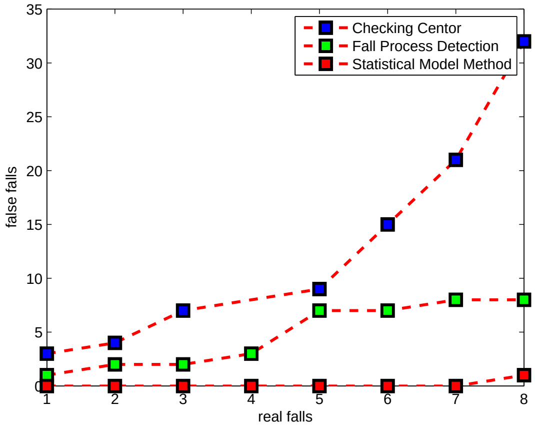

Figure 2: Detection result plot for the centroid threshold method, fall process detection and statistical model method The x-axis corresponds to the number of real falls, while the y-axis corresponds to the number of false falls. We show the detection result for these three methods with different thresholds. Every point in its dashed line corresponds to some certain thresholds.

For testing, we can compute a feature vector $[c, d, m, k]$ at each frame i. Using Bayes rule, the probability of a fall having concluded at the current frame is:

$$\begin{aligned} P(f|k,d,m,c) &= \frac{P(k,d,m,c|f) * P(f)}{P(k,d,m,c)} \\ &= \frac{P(k|f)P(d|f)P(t|f)P(c|f)P(f)}{P(k,d,m,c)} \end{aligned} \tag{6}$$

In Equation 6, all quantities can be computed based on the training data and the probability distributions that we have constructed. $P(f)$ is the probability of a frame being the end frame of a fall, which is easily computed from the frequency of falls in the training data. $P(k,d,m,c) = P(k|f)P(d|f)P(m|f)P(c|f)P(f) + P(k|\neg f)P(d|\neg f)P(m|\neg f)P(c|\neg f)P(\neg f)$, a quantity that can be determined using $P(f)$ and the conditional probability distributions we have computed for each feature for the positive and negative samples.

To determine whether the current frame is the end frame of a fall process or not, we simply compare :

$$P(f|l,t,m,c) \geq \theta \tag{7}$$

Setting different values of θ we get different rates of fall detections and false positives, as shown in the experiments.

6. EXPERIMENTS

There are 80,250 frames in our experiment data, and only eight real falls. For the threshold-based methods, we manually set different thresholds to test all of these frames. For the statistical method, cross-validation is used. We divide our videos into four groups, and use, at each cross-validation round, three of them for training, and the remaining one for testing. Each subject only appears in one of the groups, and thus testing is performed in a user-independent manner.

Because the fall action is a continuous process, if a frame i has been classified as a fall event by one of the three methods, it is

	Method 1	**Method 2**	**Method 3**
true positives	false positives	false positives	false positives
8	32	8	1
7	21	8	0
6	15	7	0
5	9	7	0
4	no setting	3	0
3	7	2	0
2	4	2	0
1	3	1	0

Table 1: Method 1 is the single-feature threshold-based method, method 2 is the multiple-feature threshold-based method, and method 3 is the statistical method. By varying the final decision thresholds, we have found settings where each method successfully detects anywhere between just one and all eight true fall events. We call "true positives" the number of true fall events detected by each setting. For each setting, we also record the corresponding number of false positives, i.e., non-fall events that were classified as fall events.

likely that neighboring frames will have similar features and will also be classified as fall events by the same method. To account for that, once a fall event has been detected, we do not allow fall events to be reported for the next 10.

The results obtained using the three methods with different thresholds is shown in Figure 2. For the second method, we always used $k_1 = 8$ and k_2 = 20, as described in Equation 5. We also give detailed results in tabular form in Table 1. We see that the statistical method has the best performance, which is an encouraging result, given the very small number of positive (fall) examples, and the user-independent nature of the evaluation of that method.

7. DISCUSSION

This paper has described ongoing work towards a system for fall detection in home environment. We have obtained encouraging results shows that vision-based methods for fall detection is promising, and that a simple Bayesian method outperforms threshold-based methods with thresholds chosen manually to fit the test data. In future work, accelerometers and multiple cameras will be employed to collect more informative experiment data. We also plan to significantly extend the size of the dataset, by recording hundreds of examples of fall actions, as well as hundreds of examples of events that can easily be confused with falls, such as bending down, sitting down, or tying shoe laces.

Acknowledgements

This work has been supported by NSF grants 0705749, 0812601, and 0923494, as well as by a UTA startup grant to Professor Athitsos, and UTA STARS awards to Professors Chris Ding and Fillia Makedon.

8. REFERENCES

[1] A.Leone, G.Diraco, C.Distance, P.Siciliano, M.Malfatti, L.Gonzo, M.Grassi, A.Lombardi, G.Rescio, P.Malcovati, V.Libal, J.Huang, and G.Potamianos. A multi-sensor approach for people fall detection in home environment. In *IEEE Workshop on European Conference Computer Vision for Multi-camera and Multi-modal Sensor Fusioin Algorithms and Applications*, 2008.

[2] A. Bourke and G. Lyons. A threshold-based fall-detection algorithm using a bi-axial gyroscope sensor. *Medical Engineering & Physics*, 30:84–90, January 2007.

[3] A. Bourke, J. O'Brien, and G. Lyons. Evaluation of a threshold-based tri-axial accelerometer fall detection algorithm. *GAIT POSTURE*, 26:194–199, July 2006.

[4] J. Chen, K. Kwong, D. Chang, J. Luk, and R. Bajcsy. Wearable sensors for reliable fall detection. In *IEEE Conference on Engineering in Medicine and Biology Society*, pages 3551–3554, 2005.

[5] B. Jansen and R. Deklerck. Home monitoring of elderly people with 3d camera technology. In *Proceedings of the first BENELUX biomedical engineering symposium, Brussels, Belgium*, 2006.

[6] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from vidoes. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008.

[7] I. Laptev and P. Perez. Retrieving actions in movies. In *IEEE International Conference on Computer Vision (ICCV)*, 2007.

[8] D. Lee. Effective gaussian mixture learning for video background subtraction. *IEEE Transactions on Pattern Analysis and Machine Intelligence (PAMI)*, 27(5):827–832, May 2005.

[9] U. Lindemann, A. Hock, M. Stuber, W. Keck, and C. Becker. Evaluation of a fall detector based on accelerometers: A pilot study. *Medical and Biological Engineering and Computing*, 43(5):548–551, October 2005.

[10] H. Nait-Charif and S. J. McKenna. Activity summarisation and fall detection in a supportive home environment. In *Internation Conference on Pattern Recognition(ICPR)*, 2004.

[11] N.Noury, P.Barralon, G.Virone, P.Boissy, M.Hamel, and P.Rumeau. A smart sensor based on rules and its evaluation in daily routines. In *IEEE conference on Engineering in Medicine and Biology Society*, 2003.

[12] N. Noury, A. Fleury, P. Pumeau, A. K. Bourke, G. O. Laighin, V. Rialle, and J. E. Lundy. Fall detection - principles and methods. In *IEEE Conference on Engineering in Medicine and Biology Society*, pages 1663–1666, 2007.

[13] C. Rougier and J. Meunier. Demo: Fall detection using 3d head trajectory extracted from a single camera video sequence. In *International Workshop on Video Processing for Security(VP4S-06)*, 2006.

[14] G. Wu. Distinguishing fall activities from normal activities by velocity characteristics. *Journal of Biomechanics*, 33(11):1497–1500, 2000.

[15] T. Zhang, J. Wang, P. Liu, and J. Hou. Fall detection by embedding an accelerometer in cellphone and using kfd algorithm. *International Journal of Computer Science and Network Security*, 6(10):277–284, October 2006.